Claims

1. A method of generating a neural network prediction model, the method comprising the steps of:-

in a first stage:

(a) training an ensemble of neural networks, and

(b) estimating a performance error value for the ensemble;

in a subsequent stage:-

(c) training a subsequent ensemble of neural networks using the performance error value for the preceding ensemble,

(d) estimating a performance error value for a combination of the current ensemble and each preceding ensemble, and

(e) determining if the current performance error value is an improvement over the preceding value; and

(f) successively repeating steps (c) to (e) for additional subsequent stages until the current performance error value is not an improvement over the preceding error value; and

(g) combining all of the ensembles at their outputs to provide the prediction model.

2. A method as claimed in claim 1, wherein the step (a) (20) is performed with bootstrap resampled training sets derived from training sets provided by a user, the bootstrap resampled training sets comprising training vectors and associated prediction targets.

3. A method as claimed in claim 1, wherein the steps (a) and (c) (32) each comprises a sub-step of automatically determining an optimum number of iterative weight updates (epochs) for the neural networks of the current ensemble.

4. A method as claimed in claim 3, wherein the optimum number of iterative weight updates is determined by use of out-of-sample bootstrap training vectors to simulate unseen test data.

5. A method as claimed in claim 3, wherein the sub-step of automatically determining an optimum number of iterative weight updates comprises:-

computing generalisation error estimates for each training vector;

aggregating the generalisation error estimates for every update; and

determining the update having the smallest error for each network in the ensemble.

6. A method as claimed in claim 3, wherein the optimum number of iterative weight updates is determined by use of out-of-sample bootstrap training vectors to simulate unseen test data; and wherein a single optimum number of updates for all networks in the ensemble is determined.

7. A method as claimed in claim 1, wherein the step (c) trains the neural network to model the preceding error so that the current ensemble compensates the preceding error to minimise bias.

8. A method as claimed in claim 7, wherein the method comprises the further step of adapting the target component of each training vector to the bias of the current ensemble, and delivering the adapted training set for training a subsequent ensemble.

9. A method as claimed in claim 7, wherein the method comprises the further step of adapting the target component of each training vector to the bias of the current ensemble, and delivering the adapted training set for training a subsequent ensemble; and wherein the step of adapting the training set is performed after step (e) and before the next iteration of steps (c) to (e).

10. A method as claimed in claim 1, wherein steps (c) to (e) are not repeated above a pre-set limit number (S) of times.

11. A method as claimed in claim 1, wherein the step (c) is performed with a pre-set upper bound (E) on the number of iterative weight updates.

12. A method as claimed in claim 1, wherein the method is performed with a pre-set upper bound on the number of networks in the ensembles.

13. A predication model whenever generated by a method as claimed in claim 1.

14. A development system comprising means for performing the method of claim 1.

15. A computer program product comprising software code for performing a method as claimed in claim 1 when executing on a digital computer.